14

Cassini star tracking and identification algorithms, scene simulation, and testing

James W. Alexander and Daniel 11. Chang
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, Pasadena, California 91109

ABSTRACT

The Cassini spacecraft uses a ~:~11-based star tracker, the Stellar Reference Unit (SRU), for attitude identification in the Attitude and Articulation Control Subsystem (AACS). Software to process SRU data resides in the Flight Computer (AFC) and is integrated with all other AACS junctions. The Cassini mission will use autonomous star identification for initial attitude determination, and a star tracking function for maintaining attitude, both performed by processing pixel data produced by the SRU and sent to the AFC via a dma interface. Because of the complexity of the StarID software, special software simulation tools were created to simulate the SRU output as a function of commands, spacecraft attitude, and star scene, and allow the introduction of fault conditions.

This paper gives the overview of the algorithm design, SRU simulation, and a description of the simulation test results and comparison with the field test results obtained using the engineering model SRU.

2 INTRODUCTION

An overview 011 the Cassini mission, design considerations for the SRU hardware and algorithms are given in refciteccassiniarch. Additional details 011 the SRU hardware are given in refcitecassinisru. For convenience, the basic. SRU characteristics arc summarized in Table 1.

The implementation of Cassini star tracking and identification functions (collectively referred to as StarID) were based on the need to support a long (12 year) mission, hardware resources, mission requirements, and previous JPL experience. To a large extent, the star identification algorithms were based 011 ground software generated to support the Astro 1 shuttle based mission (December 2-11, 1990). Most of tile initial algorithm development was in C, and tested in field tests using a JPL breadboard camera modified to functionally resemble the SRU, and finally the Engineering model SRU, which for these purposes is essentially identical to the flight units. The final StarID functions are implemented using the Ada language within a centralized 1750 16 bit computer. To support all AACS functions, the AFC has one megabyte of memory and runs at about 1.2 mips, of which the StarID function was allocated 128 kilobytes for pixel buffer storage, 50 kilobytes for star catalog storage, and uses about 3000" lines of code. A description of key elements of the algorithms, and some of the algorithm tests are given.

Because of the difficulty in scheduling field tests, and the resulting limitations of "real-sky" testing, a SRU simulation of some kind was clearly n eede d. Since a full opt ical calibration and optical testing was performed by the supplier, Officine Galileo of Florence, Italy, it, was decided that a digital simulation rather tha n an optical simulation would be the better approach, bot h in terms of fault and functional test ing, and being able to duplicate the simulation in multiple test beds where high fidelity timing was not needed. Tl If' digital only approach pod separate problems, namely how to test the final flight syst em without spec ial test hooks, and how to verify that the simulation was reasonably close what would be obtained with reel hardware, including scene and timing effects. Thus resulted the Image Emulation Unit (IEU) and the companion scene simulation software (IE U so ft ware); this combination would allow the replacement of real CCD generated pixel dat a with pixel data from a scene simulated in real-time. A sp ecial test port in the SRU (IEU port) allows the scene data to be placed into the SRU pixel data stream without test code in the flight algorithms, and *clocked* into the AFC exactly the same as if obtained from the CCD. Since the scene is synthetic the parameters can be varied to match the expected flight parameters, or to change parameters e.g., dark current, CTE, optics, or focal length, for fault testing. This approach also will allow post lau nch testing and modeling of aging effects not available with strictly hardware n iodels.

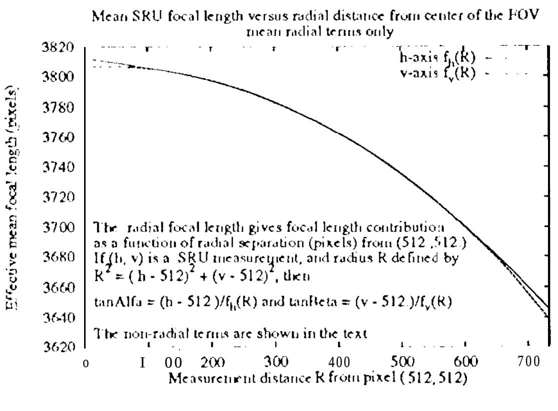
Verification of the generated scene was done by a combination of theoretical considerations and comparison to field test data. A description of the IEU software and comparison the SRU hardware is given in the second part of this paper.

2.1 Basic SRU Character istics

The Cassini SRU has no shutter or shielded frame transfer area, but makes use of the CCD analog on-chip summation modes to reduce the number of data pixels processed and to speed up the acquisition process. This also allow hierarchical acquisition of the scene within the SRU FOV. The SRU has a relatively wide FOV, the effective focal length varies with field position from 45.7mm to 43.6mm at the edge of the FOV, as shown; note that the focal length model for each axis varies separately.

Table 1. A summary of tl ie key SRU characteristics.

Function	Definition
FOV	15.6 ° x 15.6 °
CCD Format	1024 x 1024 Loral 12μm pitch
CCD Temp	Operating temperature - 35C ±5 degrees
CCD Operation	Anti-bloom ing in Partially Inverted m ode, MMP mode
Special Functions	On CCD analog summation, allowing 1, 2, and 4 pixels to be summed horizontally, 1,2,4,8,16 pixels to be summed vertically
Focal Len gth	45,7 mm (m ean)
Exposure	controllable, 1ms to 65535 ms, 400111s typical
Dynamic Range	(Software controlled)-2 mag to 6,75 for stir identi-fication, O to 6.2 mag track
Readout	1 to 5 track windows, each window specified separately
A/D	12 bit



3 StarID

The StarID function has several modes. These are summarized in Table 2.

Table 2. StarID Mode Summary

Track	Traditional star tracking mode; use 5 track windows to track 2 to 5 stars (attempts 5 stars per frame), track windows typically 5mrad x 5mrad at low S/C rates. Makes repeated measurements of the same stars, and uses ATE attitude and rate information for window placement and star verification
Mini-Reacquisition	Intermediate stage; matches stars found in enlarged track windows (2 x 2 degrees) against stars expected to be in FOV. Allows recovery from up to one degree attitude errors. Matching based on star angles, magnitude
Attitude Reacquisition	Collects star spot information using large readout areas, and matches to the star catalog. Assumes attitude error is less than 5 to 10 degrees offset from predicted boresight attitude, less than 90 degrees twist about boresight error
Attitude Initialization	This is the full "star identification" function. Collects spot measurements and matches against on-board star catalog. Uses both the position of the sun relative to the S/C to restrict the catalog search space, and provide coarse orientation information. CPU speed and memory limitations are overcome by use of sun constraint.
Telemetry	Collects spot information and computes centroids as in the attitude initialization function, but does not compute an attitude

4 STAR ACQUISITION AND IDENTIFICATION

The star identification process includes *Spot Acquisition*, (spots are SRU measurements, not necessarily identified, or even stars), setup of the catalog and measurement tables, matching a subset of the spots to stars within the catalog, and computing an attitude quaternion. Much of the implementation is designed specifically to meet Cassini mission scenarios and constraints, including those imposed by the AFC computer architecture. Since attitude can be maintained using a priori attitude knowledge, the driving factors are not extremely quick identification since this is rarely needed (minutes are allowed for AI, rather than seconds), but the necessity to avoid a false attitude identification, and the requirement to fit within the available AFC memory resources (e. g., AFC memory limitations prevented storing star-distance pairs or triangles, so restricting the set of stars to be processed in the matching is of great importance). On Cassini, there are two forms star identification.

- Attitude Initialization. The sun vector, S_{AKA}, is known in spacecraft coordinates from sensor measurements to about three degrees, and S_{J2000}, in J2000 coordinates. From this, the cosine between the SRU boresight ($+X$) axis and S_{AKA} is known, so the star catalog search can be limited to a band of stars of width (fov size +6) degrees width (21 degrees), with the midcircle of the band forming the same cosine with S_{J2000}.

- Reacquisition. The attitude of the spacecraft is limited to relatively small errors - no more than 10 degrees error in the SRU boresight vector (30 degrees will be allowed in the final version to support a specific fault protection scenario), and no more than 90 degrees error in twist about the boresight. The stars for matching are contained in a disk centered about the best guess (but wrong) attitude. The typical case will require 50 to 100 catalog stars in the star match set. Either of these constraints limits the number of stars that are simultaneous within the on-board catalog and potentially lying in a feasible FOV to be no more than about 650, rather than 3500 to 4000. The twist constraint, while not fundamental to the general reliability of the algorithm, is always available and allows some potential pairs to be thrown out early (,o speed up the process, and prevents "flipped" identification cases where the measurement accuracy may be poor.

The key steps follow.

4.1 **Spot** Acquisition. The first step is to gather a set of spots. There are several variants, but each includes taking a number of SRU frames where the images are collected using commanded summation modes, and then scanning through the pixel data to create a list of spots. The final measurement accuracy is chosen to aid in the matching process. In AI, which occurs just after launch, or under certain fault conditions, the attitude uncertainty is large, and likewise the number of stars that are included in the star match set, but the relative attitude of the spacecraft can be well controlled. Because the attitude is very stable, the accuracy of the spot measurements is made the high, which in turn is used to keep the processing time acceptable by placing tighter constraints on separation matches. in Reacquisition, where knowledge of the attitude is better, coarser measurements give enough accuracy without requiring attitude to be controlled.

SRU parameters as used for spot acquisition

Typical parameters for Attitude Initialization

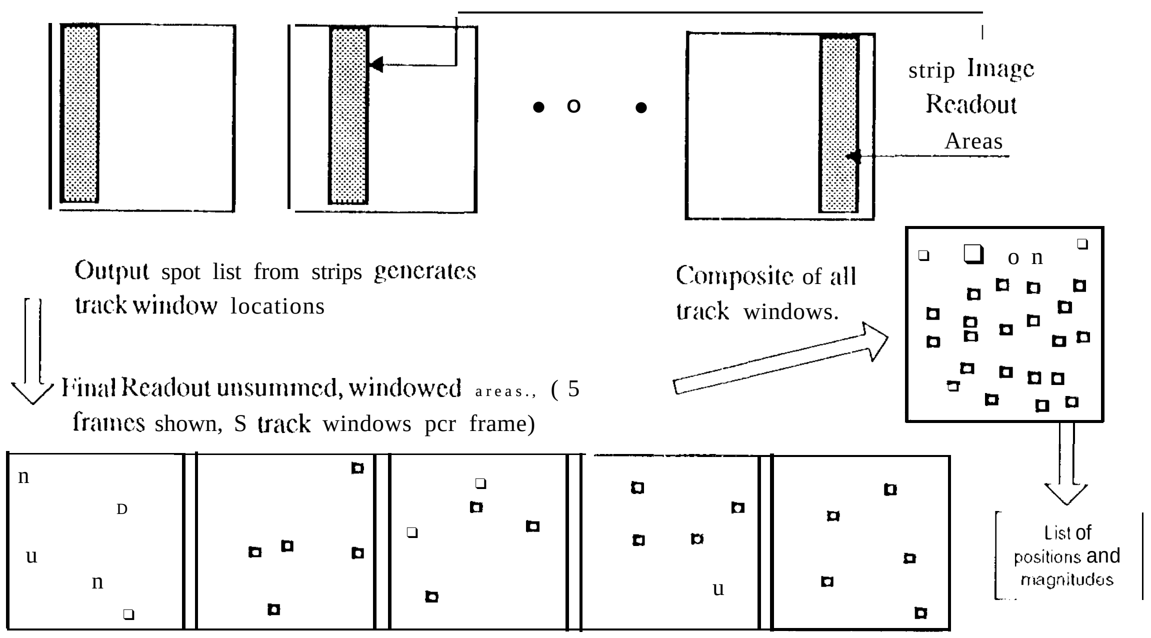
Strip Sum mode	$2h$ X 4 V
AI strips	typically 3 to9 used to cover FOV
AI "Track Windows"	One window/spot found in strips. Sum modes $1h \times 2v$ and $1h$ x $1v$
Exposure times	Fixed 1 sec. exposure used on strips. Windows 50ms to 1500ms, computed based on spots found in strip
Star search set	Aprox 21° x 360°baud, 7500 sq. deg.

Typical parameters for Reacquisition

Strips	2 to 4 strips used, "110 windows
Sum modes	$2h \times 2v$ to $4h$ x $8v$
Exposure times	5001[1s to 1000111s, depends sum mode.
Star search set	Disk centered about est. boresight, 25 to 35° diameter, (500 to 1000 deg^2), 65° for Saturn insertion support.

The FOV is partitioned into multiple large, overlapping subregions (typically 3 to 9 strips). Each strip is commanded with a fixed exposure time, (1 second for AI), using a single summation mode. Double buffering is used to shorten the acquisition cycle by processing the pixels in one frame while the next frame is taken. Each of the images is scanned for bright clusters to create a magnitude/position list. Constraints on the minimum separation between measured positions eliminates problems with extended sources, duplicated measurements, or possibly hot pixel areas. For AI (which guarantees the spacecraft to be more stable and have three degree sun knowledge), the positions of the measured spots are used to determine the placement of high resolution, smaller *Track Windows*. The the spot data measurements are much better than needed for the final identified computation, which is only used to start the tracking process, but a 100 to 200μrad/star per axis total error (.4 to .8 pixels) relative to the ensemble as a whole allows tighter tolerances to be used in the spot to star matching which in turn speeds up the processing enough to meet requirements. The spacecraft attitude estimate, based on gyro, sun sensor, dynamics model, etc., is available and is optionally (controlled by a parameter) used to compensate for image to image attitude drift, The final spots can be collected using multiple windows within 5 to 10 seconds, again double buffered. While the exact attitude is not valid, if the IRU are used, correcting the spot measurements give the equivalent of a single frame of data spots taken at the same time. Figure 2 illustrates the strips and windowing. In field tests with the engineering model SRU 20 to 35 spots are typically collected; approximately the same number as when using the scene simulations as described later.

For Reacquisition, the spots measurements from the strips are used for matching and the attitude corrections are made between the frame. The accuracy for the corrected spots should be roughly 500μrad/star per axis/ using mrad 4 × 4 superpixels.


strip Image Readout Areas
Output spot list from strips generates track window locations
Final Readout unsummed, windowed areas., (5 frames shown, S track windows per frame)
Composite of all track windows.
List of positions and magnitudes

4.2 Measurement Processing and Star Search Space Organization

'1'0 speed up processing, emphasis is placed on the measurement set, since a matching database of star separations is not available. A separation table is create for the brightest (8 to 10) measurements and up to 45 pairs of measurements. The spot measurements are corrected for optical geometric distortion, assuming a mid star color correction. For each measurement pair, the measurement numbers, (smaller number first for uniqueness), separation, an attitude angle, and a set of non-orthogonal mapping coefficients to express the constraint vectors in terms of the measurement vectors. Because the constraints are available in both spacecraft and celestial coordinates, the same set of mapping coefficients is used on star J2000 vectors to check the consistency of the star vectors with the measurement vectors for both constraints, and ultimately, orientation of triangles. The list is sorted by separation, allowing rapid lookup of whether a stars pair match some spot pair(s).

Once the measurement tables are prepared, the star matching implementation takes a straight forward and brute force approach to matching, using the pre-computed measurement tables to minimize the computation. Additionally, organization of the stars in the star search space allows some other efficiencies. Here the organization for the AI band of stars case is described, with the reacquisition case somewhat simpler. Given the attitude constraints described earlier, the sun J2000 vector and an arbitrarily chosen perpendicular vector B parameterize the search strip band. The band divided into a group of 36 adjacent partitions 'regions', each ten degrees of *clock* angle wide, and parameterized by a clock angle measured from the B vector. The region number for each star is calculated and used to partially order the set by region number - no finer ordering is used, The regions are processed by starting at the first region, processing all regions that could contain stars for a FOV centered in that regio n, then stepping through the remaining regions. For each fixed set of regions, the stars arc sequentially processed by incrementing a star counter $refstar$, and compute the pairwise distances between the star represent ed by $refstar$ and the remaining (comparison) stars within the regions, using the counter $cstar$. The sine of the pair-wise distance is computed, and compared to the sorted measured spot set. The set of all pairs containing the first star, $refstar$ is then examined for a set of stars that is consistent with both the measurements and the attitude constraints. For each star that matches $refstar$ and a pair of measurements m_R and m_C, the ordered set $\{m_R, cstar, m_C\}$ is saved, where m_R is the measurement identified with $refstar$. Note only one ordering, m_R, m_C or m_C, m_R can be accepted due to the twist constraints without the constraints, two triplets would be saved.. After processing a]] the comparison stars, we have a list of the indices representing star numbers, and measurement numbers. The list is sorted, then clustered by reference star measurements.

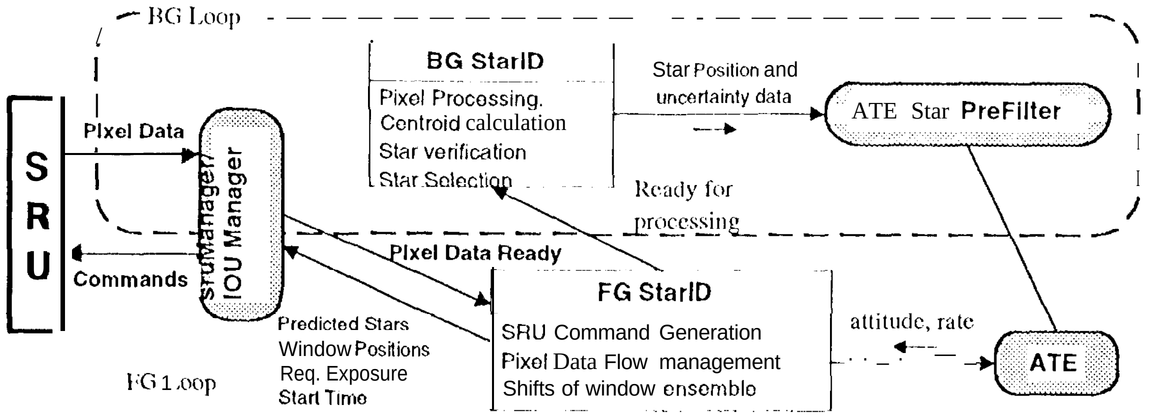
Star - Raw Spot Measurement Association])		
m_{R_1}	$cstar_1$	m_{C_1}
m_{R_2}	$cstar_2$	m_{C_2}
m_{R_n}	$cstar_n$	m_{C_n}

Star - Sorted Spot measurement association		
m_{R_1}	$cstar_1$	m_{C_1}
m_{R_1}	$cstar_1$	m_{C_2}
m_{R_1}	$cstar_2$	m_{C_3}
m_{R_1}	$cstar_2$	m_{C_4}
$\vdots$		
m_{R_n}	(-St ar_n	m_{C_n}

In the case that the number of matches is sufficient, the list of star measurement numbers to star catalog numbers are clustered, by simply grouping the stars and measurements that are consistent with the reference star. Within each set, constraints are checked and clustered into subsets that give consistent attitude quatermons. Finally, if a set of stars with at least 3 stars is found, the triple with the largest separation angle is used to compute the attitude transformation from SRU to J 2000_{guess}. At this point, we have a computed a quaternion from J2000 to the SRU frame basal on a triangle of stars that matches a triangle of measurements and meets the attitude cons trai its of the sun or other reference vector. At this point, the subset of stars neai the SRU boresight consistent with this attitude arc compared to the full set of 25 to 35 spot measurements. At this point, the problem becomes a simple magnitude and separation check between the measurements mapped to J2000. and a small set of stars. The criterion for matching now is whether enough stars match the measurement spots under the mapping. The number of stars required for declaring a match is a flight parameter. Typically five stars are required, 7 to 15 usually are matched. The attitude quaternions are computed using a modified version fo the QUEST algorithm, where equal weighting of measurements is used to reduce the storage and coding.

5 Tracking

The Star Tracking function takes more or less a traditional approach, with the possible exception of its integration with oth - FSW elements and tight interaction with the attitude est imator as shown, rather than on a stand-alone computer. Star track ing maintains the space craft att itude by making repeated measurements of known stars, by getting spacecraft attitude from ATE, choosing track stars and SRU parameters, and commanding the SRU for pixel data. The pixel data is processed, the star positions measured, verified, and the star predicted and measurement positions are returned to the ATE. If stars are lost, StarID attempts to autonomously recover by attempting star matching using the mini-reacqu isition function, or enter Reacquisition if the ATE attitude uncertainty becomes large,

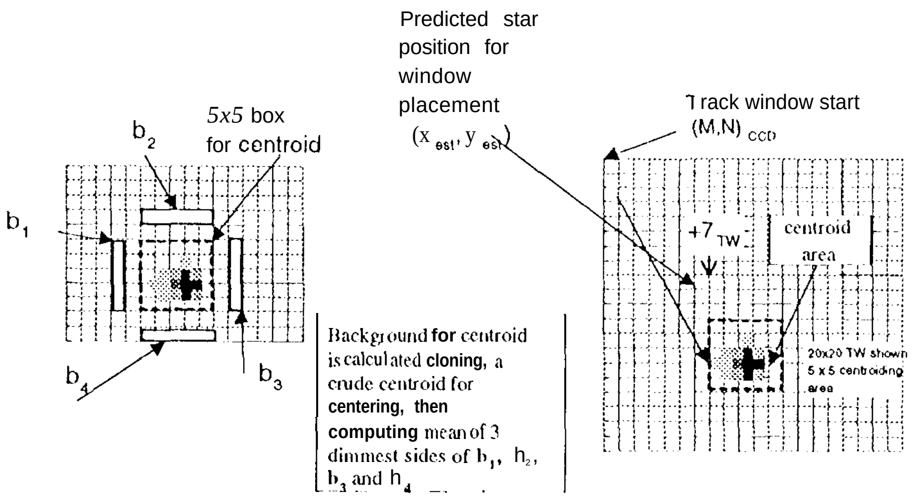


Simplified Track Interface Between StarID and FSW

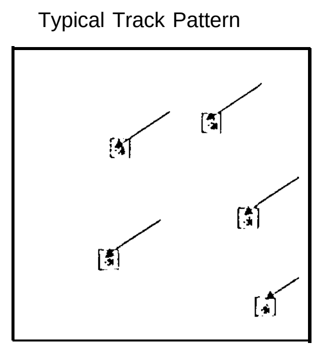
In the context of the FSW, StarID enters 'l'rack after a command to DoTrack (such as after an Attitude Initialization) or after a Reacquisition. and retains 110 information about the measurement of previous track stars.

'l'rack operates in pairs of frames, labeled as even or odd. The process is designed to be double buffered as with the attitude initialization process, with one frame commanded, while the other is being processed. '1'0 save precious CI 'U time, number of parameters are only computed or updated on the even frames. During the "even" frame, the attitude constraints, star cache constraints, window configuration, and track stars are tested and arc updated or deleted if necessary. On the odd frames, that, part of the processing is skipped except at high rate, using the same window configuration, track stars, etc.

While in track, the major data sets are the Local Star Cache (LSC) which contains all tile star-s in the neighborhood of the boresight, the Track Star List which contains the stars currently tracked, and two sets of Frame Definition parameters, one for each of the even and odd frame, and contains information about the frame timing, attitude ,and configuration. The track windows can have arbitrary placement, and can even overlap. The pixels from a commanded frame read sequentially into AFC memory. A set of simple "sorting" parameters are computed whenever the window configuration changes this presents no problem except for insuring that whenever the window configuration changes, new parameters arc computed.which allow the unsorting (only the window pixels are interspersed with pixels from another window).



Background for centroid is calculated cloning, a crude centroid for centering, then computing mean of 3 dimmest sides of b_1, h_2, b_3 and h_4



The snows point to the estimate; star in lower right hand corner would exceed error.

The centroiding assumes a locally constant background in order to keep background subtraction simple and thus is not optimized to remove streaking from interfering stars or proton events. Background subtraction removes the signal that does not come from the star of interest during tile exposure period, eliminates most readout smear, the mean dark current rate, and light from other stars or extended sources. The typical star spot against the pixel background is shown in fig. (??), which also shows the size of the centroiding window within the track window. The centroiding area is shown by the dashed box. The large track window area causes a slight . degradation in performance clue to longer readout times, but gives more timing margin in the placement of windows, and errors in the rate or position measurements from the estimator. The background DN are obtained from the neighboring pixels by summing the pixels outside the centroiding area, enclosed within the dark, rectangular area (a row or c olumn away from the centroiding area). If the centroid is not found at the estimated location, the track window is scanned for the star. All the spots found are then compared for consistency with estimate geometry. Shown to the right is a typical track frame, with the arrows pointing to the estimate. The large error in the lower right window shows a inconsistent error, so will be tossed as a valid measurement.

The LSC reduces the number of calculations required to maintain the star tracking functions, by incorporating proper motion and speed of light correction for apparent star position. It is also used for keeping a local history of which stars are successfully tracked.

When stars arc dropped, new stars arc required. however, as long as a set of track stars can be used and verified, there is no need for the reference attitude to be cllcc.keel, the LSC updated, the rate dependent track parameters checked or the track star list updated.

6 optical Model and Coordinate frame

The (3 dimensional) SRU coordinate frame is defined by the unit vectors $\{\vec{h}, \vec{v}, \vec{b}\}$ which have the nominal alignment to the Cassini Spacecraft as

Nominal Alignment	
SRU	S/C
$+h_{ref}$	+Z
$+v_{ref}$	+Y
$+ b_{ref}$	+X

A two dimensional CCD coordinate frame (h, v), (where the centroid measurements are computed) is defined by the columns and rows of the CCD, and is such that the 2-D +h axis is approximately perpendicular to the SC Y axis, and v to the S/C Z. The SRU boresight is perpendicular to the focal plane, and along the S/C +X axis.

For a pin-hole focal system with focal length f, measurements $(\Delta x, \Delta y)$, measured from the center of the FOV are mapped to the unit vector obtained by normalizing $(\Delta x/f, \Delta y/f, 1)$. The optical model used in the algorithm and simulation image model computes the distortion corrections as a position dependent focal length, described below.

Several components are used to compute the effective focal length at a centroid location. These are the temperature of the optics barrel, the star color, and the optics distortion model.

The change of the centroid position due to the optical barrel temperature is modeled by a change in the effective focal length, $\Delta f_{OptTemp}$. This focal length change is calculated based on the raw temperature measurement DN, *Temp_Meas_DN*, (approximately 31DN/degree), and is given by equation (1), with constants *Opt_Temp_Base_DN* and *Opt_Temp_to_delta_Fl*.

The star color focal length change is calculated as a function of the effective star temperature, T_e, expressed in degrees Kelvin. A color parameter, ColP is stored within the catalog, and is defined by T_e = ColP × 300. Compensation is calculated as function of the second order polynomial shown below, using (2), with stored coefficients C_h and C_v.

$$\Delta f_{OptTemp} = (Temp\_Meas\_DN - Opt\_Temp\_Base\_DN) \times Opt\_Temp\_to\_delta\_Fl \tag{1}$$
$$\Delta f_h(T_e) = C_h[0] + C_h[1]ColP + C_h[2](ColP)^2 \text{ with } \Delta f_v(T_e) \text{ computed similarly} \tag{2}$$

The optics temperature focal length correction is about 0.25 micron/degree C, so a 50 degree change causes a centroid image to shift by approximately 0.027%, and the expected color effect is about 35μrad shift at the edge of the FOV between blue and red stars, a focal length shift of about 0.013% The optical distortion correction terms that incorporate the optics temperature and star temperature focal length components are given by

$$L_0(\ldots) = \frac{1}{\Delta f_h(T_e) + \Delta f_{optTemp} + fl_h}, \quad \text{and} \quad M_0(\ldots) = \frac{1}{\Delta f_v(T_e) + \Delta f_{optTemp} + fl_v}$$

and is independent from the spot position in the FOV where fl_h and fl_v are database parameters. From the raw centroid values x_c and y_c.

$$X = x_c - 512, \qquad Y = y_c - 512., \text{ and } R^2 = X^2 + Y^2$$

$$A_h = L_0(\ldots) + L_1R + L_2R^2 + L_3R^3 + L_4R^4 \qquad B_h = (P_1 + P_3Y + P_8X^2 + P_9XY)$$
$$A_v = M_0(\ldots) + M_1R + M_2R^2 + M_3R^3 + M_4R^4 \qquad B_v = (Q_1 + Q_3X + Q_8Y^2 + Q_9XY)$$

giving the linearized form $tan(\alpha) = A_h\ X + B_h$ and $tan(\beta) = A_v\ Y + B_v$

7 STAR CATALOG

The Cassini A FC stores a full sky coverage catalog, containing data for up to 4000 stars and contains position vectors, color, instrument magnitude, and usability flags (such as the star has no similar neighbors within 0.5 degrees, or try this star first) that helps order the stars for StarID selection in choosing stars for tracking. Stars selected for the catalog are chosen as a function of position in the sky, resulting in a catalog that has different magnitude limits for different parts of the sky. As part of the selection process, it can be guaranteed that the every FOV would have sufficient coverage. The catalog storage is partitioned as declination bands of stars,sub-partitioned in right ascension regions. The starting location of each region is used an index for a star lookup function which determines the regions required to cover the SRU FOV.

Instrument magnitude is approximately $Mag_{INS} = Mag_V - 0.59\, factor \times (B-V)$

where B and V are the BVU color terms for the star.

The full on-board storage simplifies mission operations by eliminating the need to upload stars only as they are required (a traditional J] 'I, approach). Only the stars that are in this catalog can be identified or selected for track, lessing the likelihood of tracking the wrong star. The catalog will contain all stars down to about magnitude 5, and a selection of stars to about 6.5 M_I.

8 IMAGE EMULATION UNIT (IEU)

.s.1 **Testing Approach** As one of the most complex parts of the Cassini flight software, the Star]]) algorithms require a comprehensive testing approach which serves its needs while working within the framework of all flight software and subsystem integ ration act, ivities. Central to this is the Image Emulation Unit (IEU), a scene simulation package supporting each phase of StarI D algorithm development and test. The IEU is an "attitude in, pixels out" simulation of the Cassini SRU, with several hardware and software implementations. The software components are all implemented in ANSI C.

There arc. 4 distinct phases of IEU usage. These arc outlined below and illustrated in figures 1 and 2.

Unit Test The Star]]) algorithms arc prototyped in C on Unix workstations. The IEU supports this configuration by running as several Unix server processes which collect, SRU commands from StarID and return pixel readouts via the Berkeley socket protocol. Spacecraft attitude/rate are self-generated in this phase, and no real-time constraints are imposed.

As a side benefit, the c.licrlt/server protocols implemented for the Unit Test phase are duplicated in the VxWorks-based host computer for all laboratory interfaces to SRU hardware (prototypes through Engineering Model unit). This permits transparent testing of star]]) at JPL's Table Mountain Observatory facility, without software modifications to the interfaces.

Flight Software Development S ystem (FSI)S) Cassini Flight Software (FSW) is written in Ada, and the first Ada version of StarID is tested along with all other FSW algorithms in the FSDS, which is a non-real-time, software only spacecraft testbed.[7] The FSDS arch itecture is Unix based, and communicates with the FSW via Unix shared memory, using device models which closely match hardware to obviate the need for any spec ial FSW device drivers. For FSDS testing, the F 'SW Ada code is compiled to SunOS using Sun

Microsystem's Ada development package. FSDS simulates spacecraft dynamics and al] sensors/actuators. The FSDS implementation of the IEU closely resembles the Unit Test version. Essentially, the FSDS replaces the prototype StarID code as the IEU's "client," and implements FSDS-specific Pixel Output Unit (POU) and Pixel Input Unit (PIU) models to enforce timing with respect to simulation time.

Flight Software TestBed (FSTB) The Cassini Attitude- Control-Subsystem Flight Computer (AFC) is a custom built computer based on the 1750 chip set, so its availability is limited. The FSTB is developed to fill the gap between the FSDS and the Integration Test Lab (ITL). The FSTB is a real-time, Vh11;-based spacecraft testbed using a commercial 1750 single board computer (SBC) made by TASCO Inc. to host the FSW and a combination of 68040/60 SBC's and Sun Sparcstation 20's for testbed software. The 680x0 SBC's serve as the testbed host, run VxWorks as the real time operating system, and communicate with the FSW/TASCO SBC using snared VME memory. These permit a large degree of testbed code commonality with the FSDS, a parallel which is carried to the IEU. The IEU implementation for FSTB is nearly identical to that for FSDS, except for running under the Solaris 2.5 operating system. Solaris 2.5's real-time features- priority scheduling and memory-locking (to prevent memory swap-out)- are sufficient for the IEU to meet the real-time requirements imposed here. The physical interface layer between the IEU and the host SBC is Ethernet, which is fast enough with respect to both the Cassini AACS Bus and Pixel Bus to permit the use of the socket software interfaces without modification.

Integration Test Lab (ITL) The ITL is where the entire Attitude Control Subsystem is integrated and tested prior to spacecraft integration, using all relevant hardware in the loop. Thus the IEU is an integral part of the A(X SE (Support Equipment). The IEU implementation here works cooperatively with the SRU to permit fully functional closed-loop testing of the FSW "on the bench." As shown in figure 2, the Cassini SRU is designed with a test-port to accept pixel inputs from an external source instead of tile CCD A/D converter output. CCD clocking signals are sent out from the test-]~ort, for synchronization. A custom interface board, the "1EU-board", is built with a VSB (VME Subsystem Bus standard) interface on one side and SRU test- port interface on the other. The IEU software runs on an i860 array processor based SBC built by Sky Computers Inc. Flight software commands are intercepted from the AACS bus and routed to both the SRU and IEU simultaneously. The IEU uses simulated spacecraft attitude to generate the expected pixel readout, which it DMA's to the IEU-board via the VSB. To meet real- tirlle., the simulation and DMA must be performed in approximately the commanded integration time, so that as the SRU begins its CCD readout the simulated pixels can be fed to it via the test port.

8.2 **IEU Requirements** Any simulation is a balance between fidelity and speed. The capability to support real-time simulations in the ITL environment is a prime requirement of the IEU which drove many design tradeoffs, detailed later. While FSDS and Unit Test versions of the IEU do not have real-til[lc requirements, the overhead of supporting separate versions which exploit this fact for increased fidelity is excessive. Instead, the IEU is built from a common source tree to support all 4 testbeds.

The Cassini 1'1'1, simulation computers were chosen in 1991, when the i860 "SkyBolt" represented the top end of the VME single-board computer performance spectrum. The i860 is a vector (array) processor, and despite the usage of a vector compiler, substantial "hand- vectorization" of critical 1 EU computation loops is necessary to achieve sufficient speed. At the start of the IEU development effort (1993), another key constraint was that the

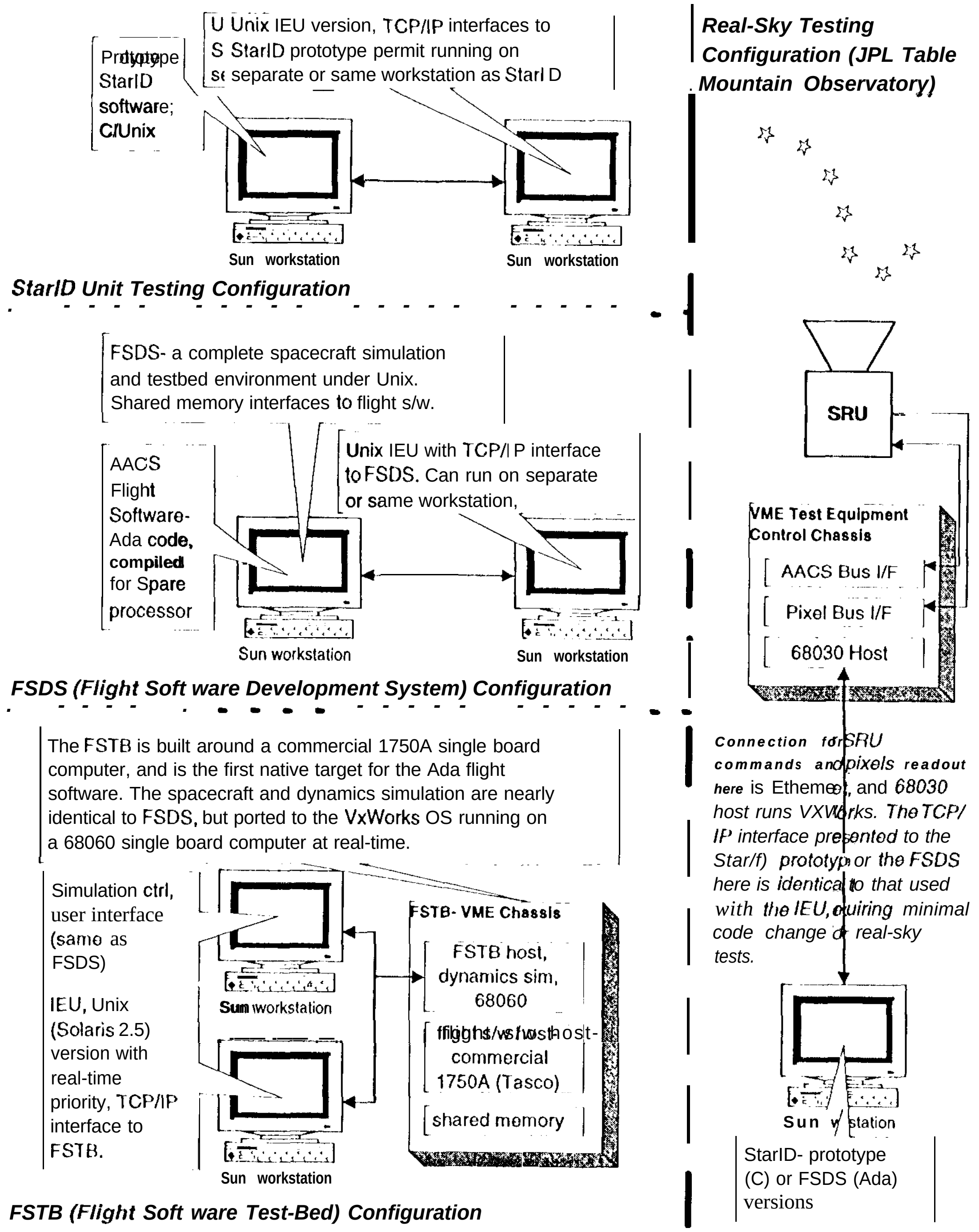

Prototype StarID software; C/Unix
Unix IEU version, TCP/IP interfaces to StarID prototype permit running on separate or same workstation as StarID
Sun workstation
Sun workstation
StarID Unit Testing Configuration
FSDS- a complete spacecraft simulation and testbed environment under Unix. Shared memory interfaces to flight s/w.
AACS Flight Software- Ada code, compiled for Spare processor
Unix IEU with TCP/IP interface to FSDS. Can run on separate or same workstation,
Sun workstation
Sun workstation
FSDS (Flight Soft ware Development System) Configuration
The FSTB is built around a commercial 1750A single board computer, and is the first native target for the Ada flight software. The spacecraft and dynamics simulation are nearly identical to FSDS, but ported to the VxWorks OS running on a 68060 single board computer at real-time.
Simulation ctrl, user interface (same as FSDS)
IEU, Unix (Solaris 2.5) version with real-time priority, TCP/IP interface to FSTB.
Sun workstation
Sun workstation
FSTB- VME Chassis
FSTB host, dynamics sim, 68060
flight s/w host-commercial 1750A (Tasco)
shared memory
FSTB (Flight Soft ware Test-Bed) Configuration
Real-Sky Testing Configuration (JPL Table Mountain Observatory)
SRU
VME Test Equipment Control Chassis
AACS Bus I/F
Pixel Bus I/F
68030 Host
Connection for SRU commands and pixels readout here is Ethernet, and 68030 host runs VXWorks. The TCP/IP interface presented to the StarID prototype or the FSDS here is identical to that used with the IEU, requiring minimal code change for real-sky tests.
Sun workstation
StarID- prototype (C) or FSDS (Ada) versions

Figure 5: Schematic of the Cassini StarID testing approach, showing how the IEU is used in 3 of 4 major testbed environments: unit test, Flight Software Development System (FSDS), and Flight Software Testbed (FSTB). Though the testbeds differ greatly, the IEU is built around same core code, and interfaces are preserved to a large extent. This has the side benefit that both the prototype StarID code and the FSDS code can be run interfaced *transparently* to SRU Breadboard and Engineering Models for real-sky testing at JPL's Table Mountain Observatory (TMO) facility.

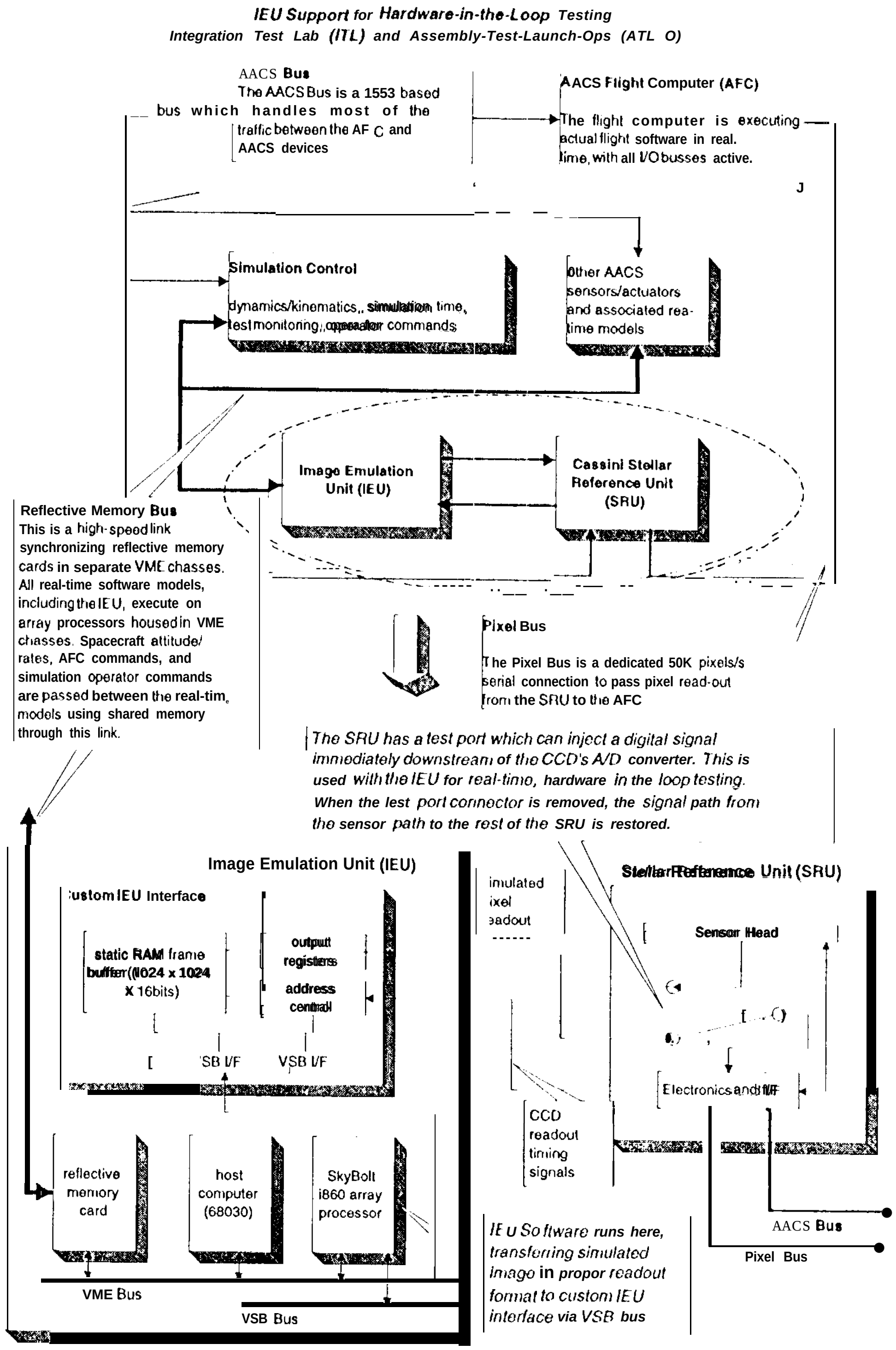

IEU Support for Hardware-in-the-Loop Testing
Integration Test Lab (ITL) and Assembly-Test-Launch-Ops (ATLO)
AACS Bus
The AACS Bus is a 1553 based bus which handles most of the traffic between the AFC and AACS devices
AACS Flight Computer (AFC)
The flight computer is executing actual flight software in real-time, with all I/O busses active.
Simulation Control
dynamics/kinematics, simulation time, test monitoring, operator commands
Other AACS sensors/actuators and associated real-time models
Image Emulation Unit (IEU)
Cassini Stellar Reference Unit (SRU)
Reflective Memory Bus
This is a high-speed link synchronizing reflective memory cards in separate VME chasses. All real-time software models, including the IEU, execute on array processors housed in VME chasses. Spacecraft attitude/rates, AFC commands, and simulation operator commands are passed between the real-time models using shared memory through this link.
Pixel Bus
The Pixel Bus is a dedicated 50K pixels/s serial connection to pass pixel read-out from the SRU to the AFC
The SRU has a test port which can inject a digital signal immediately downstream of the CCD's A/D converter. This is used with the IEU for real-time, hardware in the loop testing. When the test port connector is removed, the signal path from the sensor path to the rest of the SRU is restored.
Image Emulation Unit (IEU)
Custom IEU Interface
static RAM frame buffer (1024 x 1024 X 16bits)
output registers
address control
VSB I/F
VSB I/F
reflective memory card
host computer (68030)
SkyBolt i860 array processor
VME Bus
VSB Bus
Simulated pixel readout
CCD readout timing signals
IEU Software runs here, transferring simulated image in proper readout format to custom IEU interface via VSB bus
Stellar Reference Unit (SRU)
Sensor Head
Electronics and I/F
AACS Bus
Pixel Bus

Figure 6: With cooperative design in the SRU, the IEU is used for real-time, hardware in the loop testing in both the AACS Subsystem integration Test Lab (ITL) and subsequent spacecraft Assembly-Test-Launch Operations (ATLO). The design permits closed-loop simulations of the entire AACS flight software with no need for patching to circumvent the unavailability of a sky view.

SkyBolt relied on the use of fast static RAM's, forcing a memory limit of 8 Megabytes. It is interesting to note that, as of 1995, the HyperSparc based Sun workst at ions outperform even the vectorized i860 code by over a factor of 2.

No specific accuracy requirements were levied on the IEU itself. Instead, the general requirement is that it be sufficient to verify that St arID meets *its* well defined accuracy requirements. Verification of the IEU's accuracy against real-sky tests of the SRU En gineering Model is discussed in Section 8.5.

Similarly, 110 specific requirements on the fidelity of the IEU were levied, with tile general principle of "suflicicllcy-t o-verify-St,arII)" applying. As it turns out, this results in the IEU being a fairly comprehensive model. SRU attributes modeled to meet this principle divide into three general categories: those to test nominal Star]]) performance, those to test Star]]) robustness to SRU effects, and those to catch flight software coding bugs This is the comprehensive list model elements:

- To test StarII) nominal performance: full-sky star catalog, point spread function model, star-streaking; due to SRU motion, noise model, optical distort ion - as functions of position, star and optics temperature, windowed readout of CCD, summation modes.
- To test, StarID robustness: star streaking due to readout and lack of shutter, pixel blooming, A/D converter saturation, variable stars, proton hits, "hot" (high dark current) pixels, "dead" pixels, extended bodies in the field of view
- '1'0 catch FSW bugs using simple models: MPP mode, anti-blooming mode, vertical register gate positive voltage cont rol, firmware state machine, interface behavior

Finally, an over-riddiag design requirement is for the IEU to contain useful features to aid in algorithm development and software debugging. Much attention has been placed on this requirement, as will be detailed in Section 8.4 below.

8.3 1)escription of IEU Model Elements

Full-Sky Star catalog The star catalog for the IEU contains up to 35000 entries, reachin g into 8th magnitude. The depth is selected to adequately exercise StarID's ability to discriminate tracked stars from nearby neighbors.

The catalog is optimized for simulation needs. Each entry consists of:

- The unit vector to the star in Earth Centered Equatorial coordinates, consistent with the flight software and simulation epochs. The full vector is stored to improve search speed.
- The instrument magnitude of the star, in units of photoelectrons per second (pe^-/s) to permit simulation of the SRU's variable gain. Calculation of the instrument magnitude (Mi) from catalog visual magnitudes (Mv) uses a global fit to photometric data obtained using the SRU Engineering Model at JPL's Table Mountain Observatory, and is performed at catalog generation time to maximize simulation speed. The model is formulated as a function of cat alog B-V values; flux for a 0th Mi star is established to be 1.884×10^6 pe^-/s

A blanket scaling factor can be applied to each star's magnitude at catalog-load time. In addition, each star's magnitude can be individually adjusted at run time. These support testing of Star ID's robustness to catalog errors.

- The star color, to permit chromatic distortion simulation. The units most convenient for this purpose is Kelvins, and the conversion from catalog B-V values to K is performed at catalog generation time using a function interpolated from data in.?
- An index to a well known catalog, and various flags to provide simulation speed-up hints (for example, din-casing spot fidelity for stars beyond the flight catalog depth because they arc not expected to be tracked).

The catalog is formatted to optimize both speed and storage. The sky is divided into bands by RA and DEC (adjustable, currently 18° by 18°), with 2 polar caps of adjustable cone width (currently 15° each). The actual star catalog is stored sorted globally by DEC, then within each DEC band range by RA. The centers of each sky patch are stored along with pointers to the ranges to stars in the patch. The catalog sorting guarantees that stars within the patches can be represented by a small number of pointers to contiguous ranges in the full catalog. The retrieval algorithm then uses two dot-product searches, one for the relevant patch centers, the other for stars within the patches which fall in the field of view. This simple algorithm actually outperforms more complicated indexing schemes, as dot-product search code vectorizes well on the i860. The mean time to retrieve stars on the 40MHz SkyBolt is < 5ms, $3\sigma < 1$ms.

The catalog is packaged as a single binary file with all index and relocation data built in. This greatly reduces simulation start-up time at the expense of some possible cross-platform incompatibility. None arc identified between i860's and Sparc's; they have the same data representation and alignment.

SRU Point Spread Function A core function of the IEU is to produce a pixellated point source simulation which results in realistic. StarID centroiding performance. The IEU's point spread function (PSF) models are pre-computed and stored as tables of distributions evaluated at pixel grid points and spanning 8x8 pixels. Two ideal PSF's (a 2x2 pixel uniform and a 0.5 pixel σ Gaussian) are available along with a medium fidelity model generated from the SRU optical prescription. The ideal PSF's are useful for debugging StarID.

The JPL developed Controlled Optics Modeling Package (COMP)?,? is used to develop a set of monochromatic source ray-trace images from 0 to 7.5° incidence angle in 1.5° steps and 500nm to 900nm source wavelengths in 50nm steps, some of which are shown in figure ??. Officine Galileo provided the optical design, prescriptions, and spot diagrams to generate the COMP model. The images are pixellated at 1/48 SRU pixel sire. Note that radial symmetry is assumed in this analysis, and the products are rotated as needed (figure ??).

The monochromatic response images arc weighted using the SRU spectral response and a blackbody approximation for star spectra to generate composite response images. Analysis using these images has shown that sub-pixel centroid accuracy varies little with respect to star temperature effects on the PSF (the full range of star temperatures from 2500K to 40000K was considered). The conclusion is that the PSF's generated for IEU use need consider only one representative star temperature, currently set at 6000K. Note this is a separate effect from star centroid shifting due to chromatic distortion, which is significant and is modeled separately (see page 17).

The detector plane is divided into 128x128 pixel regions, and the composite responses arc rotated as appropriate to the regions' centers. The nearest two radial incidence angle images are interpolated to form the images for that region. Each image is re-centered according to its centroid to compensate for roundoff in the rotation and interpolation. Finally each image is normalized to unit magnitude and numerically integrated to produce proper distributions, discretized to 0.1 pixel spacing. This process is illustrated in figure ??. The end product is a 2 dimensional table of PSF distributions as functions of row and column. These are packaged along with the two ideal PSF's into one binary file.

The choice of using an ideal or modeled PSF is selected via user command at run-time. An additional fidelity level is defined- stars which fall in a commanded readout window (the only ones which StarID cares about) are calculated over a 6x6 pixel region, which others get a 2x2 pixel spot (mostly for purpose of display- see Section 8.4).

At run-time, star center positions on the detector plane are generated from the optical distortion model (see below). The PSF distributions at each pixel corner of the surrounding 2x2 or 6x6 pixel region is evaluated from the stored tables using nearest approximation or bilinear interpolation respectively, permitting calculation of the total energy received at each pixel using 4 sums. The pixel values are evaluated in readout data-number units (*dn*), which scale from the star's instrument magnitude through the commanded SRU gain. Recall that the star's instrument magnitude is stored in the simulation catalog, and that the PSF itself is unit magnitude.

Star Streaking due to SRU Motion Spacecraft motion clearly has an effect on the star image and the resulting StarID centroid accuracy. Since star tracking involves snort (almost always < 1s) exposure times, and since the IEU must meet real-tillle requirements, the spacecraft rate is assumed to be constant for each exposure.

The computation of the motion streak has the greatest speed impact of all algorithms in the IEU. Therefore, a simple Euler integration model is used in which the streak is divided into individual still sub-exposures and summed. This direct algorithm has 2 major advantages: it is easy to vectorize for speed on the i860, and it permits easy tradeoff between fidelity and speed 011 a single parameter- the number of sub-exposures per pixel of star motion.

Stars are categorized as one of four fidelity levels at runtime to maximize simulation speed. No motion streaks are computed for a star outside of commanded readout window. A still 2x2 pixel spot is generated at the mid-exposure position instead, mostly for display. No motion streaks are computed for stars which are in readout windows but dimmer than 6.5 Mi, as they are not expected to be tracked. The fidelity levels for the remaining stars depend on the rate- exposure product, which determine whether a floating point accumulator is used. To conserve memory, pixels values are stored as unsigned 16-bit integers. While this gives sufficient dynamic range, it, results in truncation errors in the summation of the sub-exposures. When the rate- exposure product is below 36 pixels at the star's location on the detector plane, the streak is constructed in a floating point accumulator, then transfered to the frame buffer, resulting in greater accuracy, 36 pixels subtend the streak produced by a $2s$ exposure at $0.25°/s$ of transverse motion (rate component about the boresight transform to slower pixel rates given the limited field of view), easily enough to cover the range of applicable StarID accuracy requirements. Stars in the highest fidelity level use 3.0 sub-exposures per pixel; the others use 1.5 or none.

SRU Noise Models Proper modeling of noise is crucial to extracting the correct performance from StarID in testing. Three sources are considered: readout noise, dark current, and photon shot noise. The basic model for the noise at each pixel is the following:

$$N = \mathcal{N}(0, K\rho) + \mathcal{P}(KT\mu) + \mathcal{P}(Ks) \quad (3)$$

where N is the noise value in dn, $\mathcal{N}$ is the normal distribution, $\mathcal{P}$ is the Poisson distribution, T is the exposure time [see], K is the SRU gain [dn/pe^-], μ is the dark current rate [pe^- /s], ρ is the readout noise σ [pe^-], and s is the mean pixel signal [pe^-].

However, running random noise generators of the appropriate distributions on-line to calculate equation 3 while meeting real-time requirements is impossible. Instead, the following concessions are made:

- Noise is computed only for pixels in readout windows.
- The Poisson processes- shot noise and dark current- are approximated as Gaussian with appropriate mean and variance. Dark current variance is modeled to vary linearly with integration time, and photon noise variance linear with respect to star signal. Note the latter assumes a properly designed ("photon limited") signal processing chain in the SRU. Readout noise is modeled as zero mean Gaussian.
- The use of Gaussian distributions permit the pre-computation of large "base" vectors at simulation startup, leaving only biasing and scaling to be done in real-time. These operations vectorize well on the i860. Three Gaussian base vectors, each 32768 elements long, are pre-computed and stored: two for readout noise with variance consistent with the SRU's two gain settings, and one of 0 mean and unit variance. Since the maximum number of pixels read out per frame is only 1024, selecting a uniformly distributed random starting index every frame for extracting noise vectors gives a reasonable approximation of uncorrelated noise from frame to frame.
- For large readout windows, the readout time can be long with respect to the integration time. Therefore the last pixels can accumulate significantly more dark current than the first. This dark current gradient is ignored. The impact is minimal, as high accuracy track situations all involve small windows.

Optical Distortion Model The SRU optical distortion correction model is developed by Officine Galileo. It is formulated as two 9th order polynomials, and take into account position in the field of view as well as star and optics barrel temperature:

$$\tan(\alpha) = p_{x1} + p_{x2}x + p_{x3}y + p_{x4}xr + p_{x5}(r^2) + p_{x6}x(r^2)^{1.5} + p_{x7}x(r^2)^2 + p_{x8}x^2 + p_{x9}xy \quad (4)$$
$$\tan(\beta) = p_{y1} + p_{y2}y + p_{y3}x + p_{y4}yr + p_{y5}(r^2) + p_{y6}y(r^2)^{1.5} + p_{y7}y(r^2)^2 + p_{y8}y^2 + p_{y9}xy \quad (5)$$

where $\tan(\alpha)$ and $\tan(\beta)$ are *defined* to be the ratios v_x/v_z, v_y/v_z of the measurement star vectors in the SRU reference frame respectively, x and y are positions from boresight location on CCD (units depend on units of coefficients- IEU uses pixels), $r^2 = x^2 + y^2$, and the p_{x2}, p_{y2} terms contain the temperature dependence according to:

$$p_{x2} = \frac{1}{c_{x0} + c_{x1}T_e + c_{x2}T_e^2 + 0.001(k_{11} + k_{12}T_o)} \quad (6)$$

$$p_{y2} = \frac{1}{c_{y0} + c_{y1}T_e + c_{y2}T_e^2 + 0.001(k_{t1} + k_{t2}T_o)} \quad (7)$$

where T_e is the star temperature in Kelvins, and T_o is the optics barrel temperature measurement in ° Celsius. Note equations 6 and 7 capture the effects of chromatic distortion and optical barrel thermal effects by stretching the effective focal length.

Note these equations are formulated for use in StarID, to correct $(\tan(\alpha), \tan(\beta))$ given the centroid positions. The IEU needs the inverse of this: given the true tangents as computed from the attitude and catalog, predict the star centroids on the detector plane. Rather than solve for (x,y) in the full nonlinear equations analytically, a fast iterative method is used. Equations 4 and 5 are rewritten as:

$$\underline{z} = [A]\underline{x} + \underline{h} \quad (8)$$

where:

$$\underline{z} = \begin{bmatrix} \tan(\alpha) \\ \tan(\beta) \end{bmatrix} \quad [A] = \begin{bmatrix} p_{x2} & p_{x3} \\ p_{y3} & p_{y2} \end{bmatrix} \quad \underline{x} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \underline{h} = [(H.O.T)]$$

Recalling this is the simulation, where the '(truth') position of the star is available, equation 8 is iterated using the true tangents for $\underline{z}$:

$$\begin{aligned} \underline{x}_{k+1} &= [A]^{-1}(\underline{z} - \underline{h}_k) \\ \underline{h}_{k+1} &= \mathcal{F}(\underline{x}_{k+1}) \end{aligned}$$

where $\mathcal{F}$ is the evaluation of the higher-order-terms in equations 4 and 5. The iteration starts using (0, 0) for (x,y) and continues until a per-step RSS tangent delta of $< 1.0 \times 10^{-8}$ radians or a maximum iteration count of 10 loops is reached. It is found that the error threshold is reached in at most 5 iterations at the field of view edges, and that the algorithm does not converge to local minima. The execution time is also minimal: less than 0.2 ms to converge on the SkyBolt.

The distortion model is turned off simply by skipping the iterations, a useful capability in debugging. The model as delivered by OG and its inversion in the IEU have been roughly verified against observatory test data gathered with the SRU Engineering Model- see Section 8.5.

Windowed Readout, Summation Modes To increase signal to noise ratio (at the expense of accuracy) for high rate or dim star situations, the SRU can be commanded to perform analog summation in both line and column directions before A/D conversion. In addition, readout of up to 5 windows of arbitrary geometry and overlap can be specified, with the restriction that window boundaries must be consistent with the summation modes or on 4-pixel boundaries, whichever is more restrictive. The potential overlaps can result in substantial geometric complexity which StarID must properly handle to extract centroids from the readout memory image. The IEU independently models these SRU features to permit full regression testing of StarID.

Star s treaking due to Readout The SRU has no shutter, and the entire CCD is exposed to maximize accuracy. Therefore, stars are exposed during readout, resulting in streaks. In addition, windowing the readout can result in "disconnected" streaks "below" each window. This effect is caused by the large disparity between the readout time for a pixel requiring A/D conversion (20 $\mu s/pixel$) and one which does not (0.6 $\mu s/pixel$). As an image is shifted off the CCD, pixels are subject to uneven exposure to star light depend on whether pixels "before" are subject to A/D), resulting in disconnected streaks. This is shown in figure ??, which is a contrived example in that the readout, windows shown should never be requested by StarID, but does ~r-lake the geometry of the effect clear.

StarID is designed to reject both connected and disconnected streaks in its spot finding functions. To verify the design, the IEU models both effects. The geometry of the model is complicated in detail but conceptually straightforward. Two simplifications are assumed: lines with no pixels requiring A/D conversion are shifted instantaneously (it. no exposure), arid the streaks are fixed width (4 pixels across) using a 1-dimensional Gaussian point spread function (σ = 1 pixel). Spacecraft rates during readout can result in slanted streaks- these are taken into account as WC]]. The spacecraft rate is assumed to be constant during the readout. This becomes unrealistic for large readout windows approaching the full CCD (20s readout), but is fine for windows typical of StarID operation, and results in significant algorithm simplification.

The geometric fidelity of this model was verified by inspection against real-sky images collected using breadboard and Engineering Model versions of the SRU.

Pixel Blooming, Anti-Blooming Mode CCD pixels which receive more exposure than full well can bloom. This saturation] at the top of the device's dynamic range can significantly affect StarID centroid accuracy, and thus must be modeled in the IEU. The model algorithm is simple enough to run fast, while achieving a realistic centroid degradation, which is its purpose:

- The assumption is made that overflowing charge affects pixels along the readout direction only ('(rows"), reducing the model to one dimension. This is consistent with the SRU CCD's sub-pixel structure.
- The PSF model builds a list of all over-cxj,oscci pixels as it generates spots. The full well value is a parameter established through SRU testing. The "extra charge" is stored temporarily in each simulated over-exposed pixel.
- The bloom model is run immediately after the PSF model, but before the addition of noise. This simplifies the process of finding spot boundaries.
- For each pixel in the over-exposure list, the spot boundaries along the readout direction are determined, and a 1-D centroid is performed. The charge above the full well limit is summed for all the pixels between the boundaries, and each pixel is removed from the over-cx~,osurc list to eliminate redundant processing. This sum represents the total "extra charge" to be distributed.
- The "extra charge" is divided in half and spread evenly up and down from the centroid pixel. Each pixel's value is increased up to full well until the extra charge is depleted. This results in the characteristic bloomed-spot appearance.

  Note this model conserves the total charge resulting from the exposure, which approximates physical reality. The results are fairly realistic- see figure ??

'['he SRU has an anti-blooming mode. The effectiveness of this design is established through testing.? The IEU's intent in modeling this is only to catch flight software bugs, using as simple an algorithm as possible. over-exposed pixels are replaced by pixels at full well, essentially modeling perfect anti-blooming.

A /D Conversion, Gain The SRU has 2 gain levels: 30 pe^-/dn in low gain, and 10 pe^-/dn in high gain. The CCD's full well level is approximately 100000 pe^-, and the A/D converter is 12 bit (0-4095 dn). Therefore, at high gain, A/D saturation is reached before full well. In addition, the DC offset on the A/D conversion is selectable, discretized to 8-bits. The offset is analog; therefore, the mapping to dn's is a function of both gain and summation mode settings, These effects are properly accounted for in the IEU.

Variable Stars Failure to account for variable stars have caused troubles with star identification algorithms in previous missions. While the Cassini StarID approach depends only peripherally on magnitude information and is therefore less immune to this effect, the immunity still needs to be verified. Since the brightness is essentially constant for time-scales which matter for identification, the IEU simply permits individual stars' magnitudes to be varied via simulation console control. This avoids the needless complication of tying these test scenarios to any celestial reference time.

Radiation Effects Three radiation effects which can significantly affect StarID accuracy are modeled straightforwardly in the IEU. Proton impacts are modeled as transient flashes of constant brightness (3000 dn). Over the course of the long mission, pixels may develop excessive dark current rate ("hot") or become insensitive to light ("dead"). Dead pixels are modeled using a constant dn. For simplicity, not pixels are modeled as having a user specified flux rate, effectively increasing the mean of the dark current, noise model for that pixel while leaving the variance unchanged. The user can construct files specifying hot and dead pixels, and update them through the simulation console. In addition, the user can specify either random or deterministic moving patterns of proton hits, The emphasis is on repeatability, to aid testing.

Extended Bodies Since Cassini will be operating in the vicinity of Saturn and its moons, the possibility of star occultation exists. In addition, an extended body can confuse StarID's spot finding functions. To support real-time operation, the IEU incorporates an extended body model which captures only the gross effects important to StarID. A body consists of a sphere with 2 constant regions of brightness separated by a terminator. Up to 16 bodies may be modeled simultaneously, including the Sun as the sole illumination source. The bodies' positions are specified externally and may be arbitrary or consistent with JPL generated ephemeris data files for various mission phases. Figure ?? shows an arbitrary arrangement of 2 extended bodies in front of the Orion constellation.

The simulation does not render the extended bodies through the same high fidelity optics model used for point sources. In addition, spacecraft motion smearing of the extended bodies is ignored. A simple algorithm is used to capture the gross effects of readout smearing- essentially blanketing the pixels below the body. These concessions permit meeting real-time requirements.

Other SRU Modes The SRU has a command-able Multi-Pinned Phase (MPP) mode, which is a technique to suppress dark current. ? The SRU implementation affects the full well level only minimally. Therefore, the

IEU model for MPP is simply to select a different dark current rate.

The positive voltage used to effect pixel shifting for readout is adjustable, to compensate for anticipated effects of long term radiation exposure.? The effect of incorrectly setting this control is to greatly decrease the full well level. Since the purpose of modeling this effect is to catch flight software bugs, the IEU simply reduces the full well level in the blooming model by 80% when the commanded value differs by more than 5 *dn*'s from the correct one, resulting in an obvious effect. A warning is printed as well.

SRU State Machine Figure ?? shows the SRU mode transition diagram. In normal operation, StarID uses only STANDBY and INTEGRATION & READOUT modes. CLEAR is an initialization mode which exits autonomously, and FLUSH/TEST modes are convenient for lab testing.?,? The IEU models the full state machine in order to trap flight software bugs (commanding into TEST mode, for example). However, SRU outputs for the FLUSH and TEST modes are not modeled.

Interface Behavior Cassini carries 2 SRU's and 2 flight computers for redundancy. Each SRU has 2 data interfaces to each flight computer: the AACS Bus, a modified 1553 bus which provides command packets, and the Pixel Bus, a custom serial interface providing the readout.? Both interfaces are cross-strapped. AACS bus packets provide values for 33 8-bit registers which form the command and status interface for the SRU. To test the validity of flight software, the IEU software fully decodes the AACS bus packets and maintains an internally consistent set of SRU register values. Cross-strapping behavior of the interfaces, significant only in the ITL test environment (figure 2), is simulated to a limited extent. Dictated by the larger ITL architecture and the desire to minimize cost, a single copy of the IEU software running on one SkyBolt simulates both SRU's, taking; advantage of the fact that they are not used simultaneously. Two custom IEU interface cards receive simulated pixels from the one SkyBolt, distinguished by different VSB addresses and providing the cross strap to the rest of the system. AACS bus packets for both SRU's are intercepted and sent to the IEU.

8.4 IEU Features Useful for StarID Debugging As the IEU is the primary testbed for the development and verification of StarID, its usability as a debugging tool is as important as its fidelity as a model. The features described in this section have proven useful in the course of the Cassini project, and should apply to similar future efforts.

Scene Displays - Modified SAOImage By far the most useful IEU feature is the incorporation of a scene display in all its incarnations. The display is implemented such that real-time operation is largely uncompromised where needed, in every version, the option to display to a modified version of SAOImage exists, In the ITL version, the operator may choose do display to a VME memory mapped display card or to SAOImage.

SAOImage was selected as the display tool for several reasons: it's freely available with source code, in wide use in the astronomical community, has a reasonably good user interface to its feature set (zoom, pan, color-map manipulation, etc.), and is by our ad-hoc testing one of the fastest pseudo-color display programs freely available for the Unix/X environment.

The various I/O options for the basic SAOImage (we used version 1.07E2) - file, TCP/IP socket, IRAF Imtoolwere found to be too slow for our specific purpose. Therefore, SAOImage was modified to use a Unix signal and shared memory interface with the IEU. After generating the scene, the IEU sends a signal (Unix SIGUSR1) to the modified SAOImage, which copies and displays the scene, made accessible via shared memory. In the Solaris 2.5 version of the IEU, memory locking is utilized to prevent the snared memory segments from being swapped. The IEU/display handshake is such that the display never holds up the IEU. An additional feature found indispensable in the Star ID unit development phase is the drawing of readout, windows borders and crosses on the display representing a star's true position, StarID's measured centroid, and the predicted position of the centroid using Attitude Estimator data. Many bugs and design flaws were found by observing the patterns which these quantities made, with close-up observations being possible using SAOimage's zoom feature. The window borders are included in the shared memory interface, while cross coordinates are passed to the display from the individual sources via datagram sockets (which are available for the C/Unix unit test versions of flight algorithms). Loose synchronization in time was found to be sufficient.

Figure ?? shows a sample of the IEU SAOImage display

In the ITL environment, where the IEU runs on the SkyBolt single board computer, the scene display interface is more complicated. The Sky Bolt has no on-card network interfaces, relying entirely on DMA transfers on the VME bus. For a basic "quick and dirty" display, a DMA dump to a VME video card is available via simulation console command. In addition, the IEU can also DMA its readout to a relay program running on the host single board computer which has an Ethernet interface. The relay passes the pixels to SAOImage running on a workstation via a socket interface. Because of the comparatively slow Ethernet link, this display option is only practical for small readout windows (small pixel count) characterizing star tracking situations. The user interface advantages are significant, however.

It is ironic that the ITL-SAOImage display arrangement has found considerable use in actual SRU testing at JPL's Table Mountain observatory, where the SRU is controlled by the same type of VME single board computers, permitting live-sky displays of the prototype StarID in action with almost no software modification. The live sky tests emulate the simulation well.

Simulation Control A key to successful debugging is the ability to turn of imperfections to gain confidence in the test algorithm incrementally. Every effect in the IEU - PSF, optical distortion, noise, readout streaking, blooming- can be adjusted to approach an ideal tracker simulation. Even the effect of pixellation can be removed by removing StarID altogether from the loop and passing truth measurements directly to the flight software, disguised as an image readout. Dubbed the "spoof mode," this option proved useful for Attitude Estimator debugging.

A user console with scripting ability for simulation control proved to be a necessity. The IEU supports a simple string based commanding scheme. For all Unix incarnations these were embedded into the TCL/Tk framework. The ITL supports its own console to pass in the same commands.

Debug Printouts The print statement is the most primitive of debugging aids. However, in the limiting environment of the real-time testbeds and the flight computer, printouts from the IEU provides invaluable visibility into the flight software. The user can select from the simulation console printouts of all important quantities available: StarID commands, simulated SRU register values, truth attitude and propagation data, all I/O activity for hardware-in-the-loop debugging, model execution time estimates, and catalog data on stars in the field of view.

Integrated Parameter Files Management of IEU parameters to effect testing scenarios is tedious and prone to error. A facility to group all parameters into a single file which can be given an intuitive name has proven useful.

8.5 **IEU Model Verification** The majority of the IEU's complexity is in external interfaces and models of SRU functionality. These are verified by inspection and many small tests in the course of software development, and will not be discussed further. Instead, the focus in this section is on end-to-end verifications of the IEU's accuracy, which involve both stand-alone tests and tests run with StarID, using data collected with the SRU Engineering Model. These tests provide the confidence necessary for the IEU's to be used in integrated flight software testing.

Polarity A basic but important verification is that of polarity. It was demonstrated that when StarID's identified attitude using a real-sky image is given as input to the IEU, the resulting simulated image matches the real-sky image. In addition, the images were verified by inspection of the sky to conform to the SRU mechanical interface specifications. This verifies the chain of transformations involved.

Spot and Streak Centering The many steps involved in generating a star's image are prone to introduction of error, both in the off-line PSF model generation and the on-line computations. Since the "true" star centers are known to the simulation, a simple verification is to compute centroids from the simulated image and compare. Figure ?? shows a typical result, for a particular field of view with a wide range of star brightness. To isolate error contributions, all effects (noise, readout streak, etc.) are turned off except for streaking due to spacecraft motion. The resulting accuracy should therefore exceed expected SRU performance, and they do. The rate-exposure product (streak length) in figure ?? is approximately 10 pixels (0.1 <2 °/s for 1 sec. exposure, zero component about boresight). It is observed that, given this artificial case of no background noise to degrade the signal to noise ratio, the fairly long 10 pixel streak improves centroid accuracy to the 1/100th pixel level, an expected result, which further indicates the streak-readcrirlg part of the IEU works properly.

PSF Model Accuracy (fill me in!)

Distortion Model Accuracy The accuracy of the distortion model is verified at two levels: stand-alone verification of the inversion algorithm implementation described on page 17, and verification of the OG-delivered coefficients themselves in conjunction with StarID, using real-sfiy images collected with the SRU Engineering Model. The sta~ld-alone verification involves simply comparing truth' star tangents (known to the IEU) with results of independently applying equations 4 and 5 to the post-distortion-model star centers. Figure ?? shows the resulting tangent differences are on the order of the model's inversion tolerance (1.0×10^{-8} radians, or 1/25000 pixel), showing that the model implementation itself contributes negligible error. The real-sky verification compares the raw centroids acquired by StarID with the IEU-predicted locations of those spots, with the IEU using as input the attitude estimate calculated by StarID). Figure ?? shows the results using one frame acquired on 9/27/1995. The typically achieved accuracy from that observation run is about 0.5 pixel.

This is the expected accuracy level for the conditions of the particular test: through air, no optical-barrel temperature correction, and StarID using the reduced-accuracy (and faster) centroid algorithms present in the Attitude Initialization mode.

Magnitude Accuracy Recall that the IEU's catalog stores instrument magnitude. Therefore, there are two parts of "magnitude accuracy" for the simulation:

- Properly mapping the stars' visual magnitude to instrument magnitude for the SRU.
- Properly generating a spot with the expected DN values.

Preliminary analysis for the first part using lab and real-sky data has been performed.[7] Plans call for revisiting 'Table Mountain observatory to gather additional data using the SRU Engineering Model to finalize this analysis, which is needed for both flight and simulation catalog generation. The second part is simulation-specific, and is verified in cooperation with StarID. StarID independently estimates the instrument magnitude of a tracked star given gain, exposure time, etc. It is observed that StarID's magnitude estimates consistently match the IEU's catalog inputs, implying that the IEU's intermediate chain of calculations are correct The less likely scenario that they are both independently and symmetrically wrong is ruled out by inspection.

9 ACKNOWLEDGMENTS

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

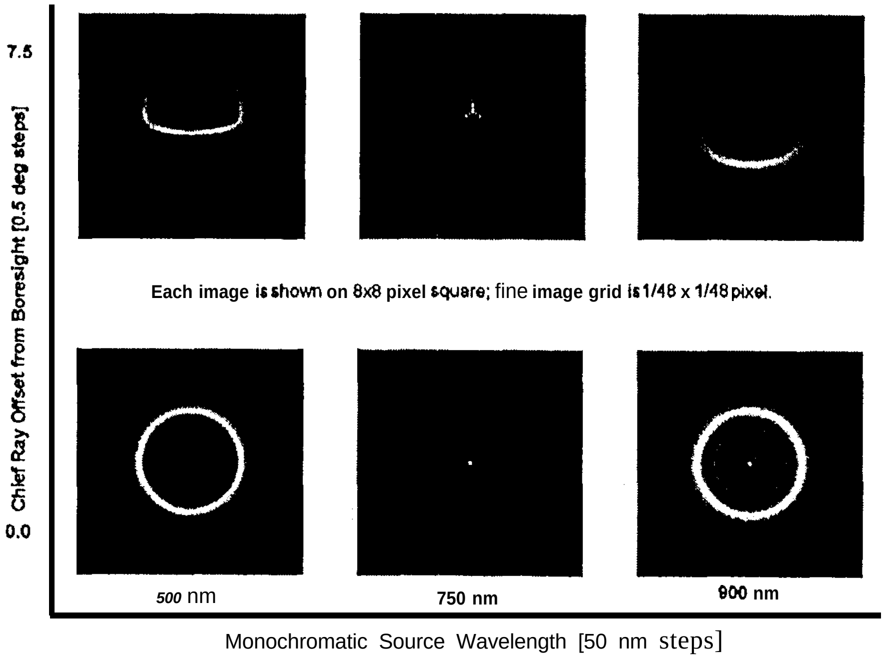
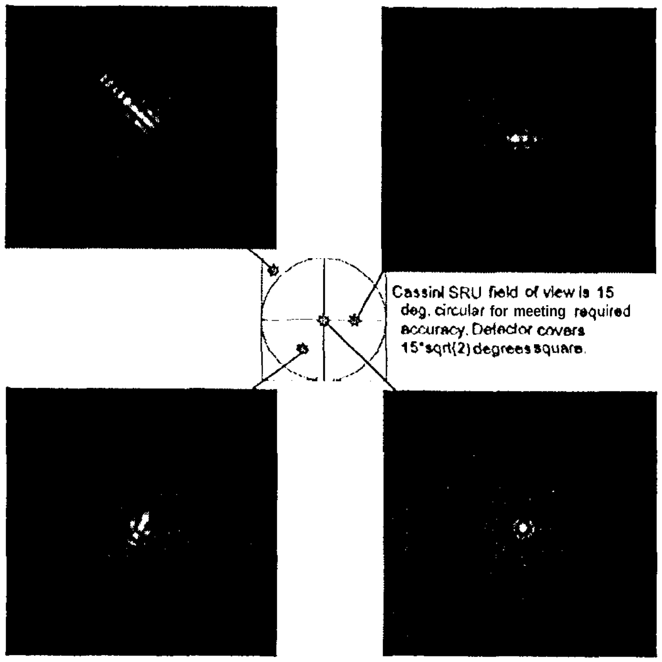


Figure 7: COMP-generated monochromatic responses as functions of wavelength and radial position on detector plane. These form the basis for the IEU's PSF model.



Cassini SRU field of view is 15 deg. circular for meeting required accuracy. Detector covers 15*sqrt(2) degrees square.

Figure 8: Sample PSF's, created from weighted monochromatic responses and rotated/interpolated to represent positions on the detector plane.

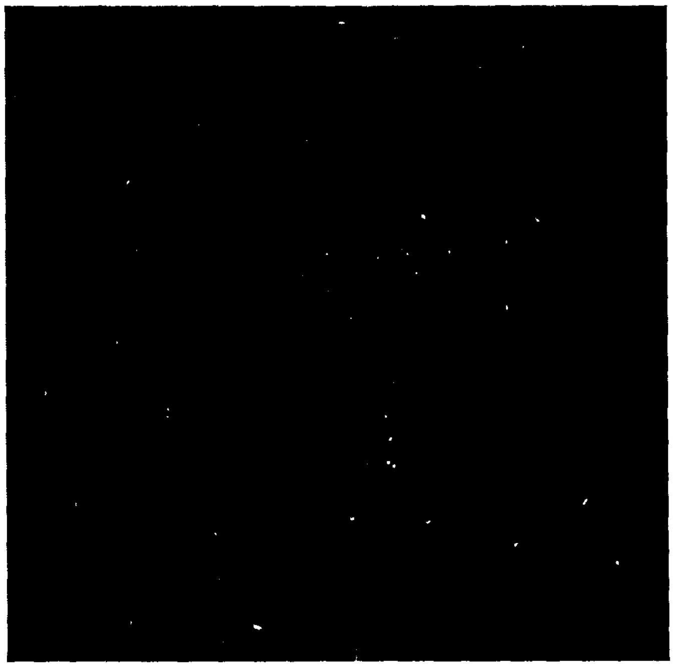


Figure 9: Example showing IEU's model of star streaking due to spacecraft motion. The spacecraft rate vector is purely along the SRU borsight (in this case 10/s for a 1 sec. exposure). For clarity, both readout streaking and readout noise models arc turned off.

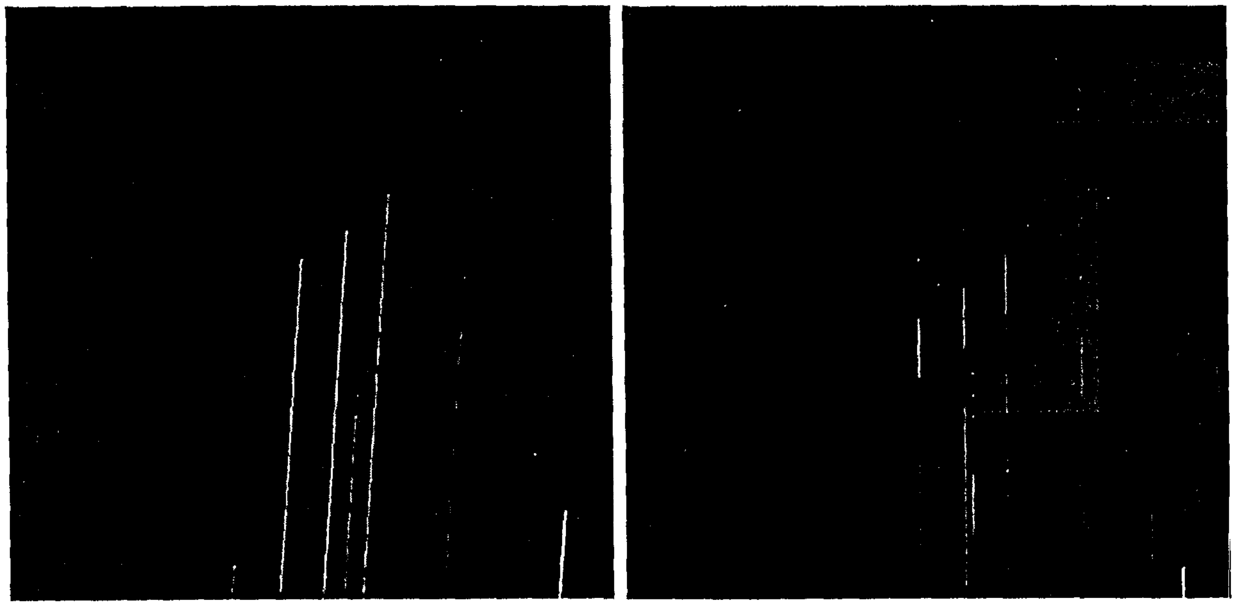
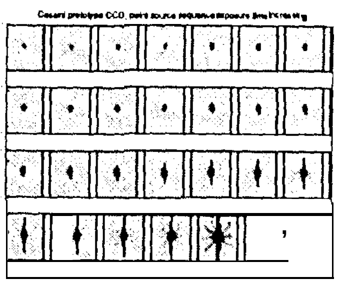


Figure 10: Two examples of the IEU's model of readout streaking, showing "connected' and '[cl isconnected" streaks. Streaking occurs because the CCD is exposed during frame readout (the SRU has no shutter). Disconnected streaks result from readout window geometry and the fact that pixels requiring A/D conversion are processed much slower than those which don't. The connected streak example also shows that spacecraft motion is taken into account, in this case a 0.050/s transverse motion. Note the apparent "highlighting" of the readout windows is a result of noise being computed only for window contents.



Pixel Blooming Simulation

- Pixels blooming upon reaching full-well limits the dynamic range of the star tracker and therefore affects star tracking performance. IEU simulates blooming when SRU anti-bloom mode is off

- Lab data shown at left taken using prototype SRU CCD.

- Simulated images below of Southern Cross, ~ 6 by 6 deg. square

- Separate issue: since this readout window is near center of detector, the bright stars In the widow do not show readout streaks, but note the presense of stinks due to bright stars "above" the readout window.

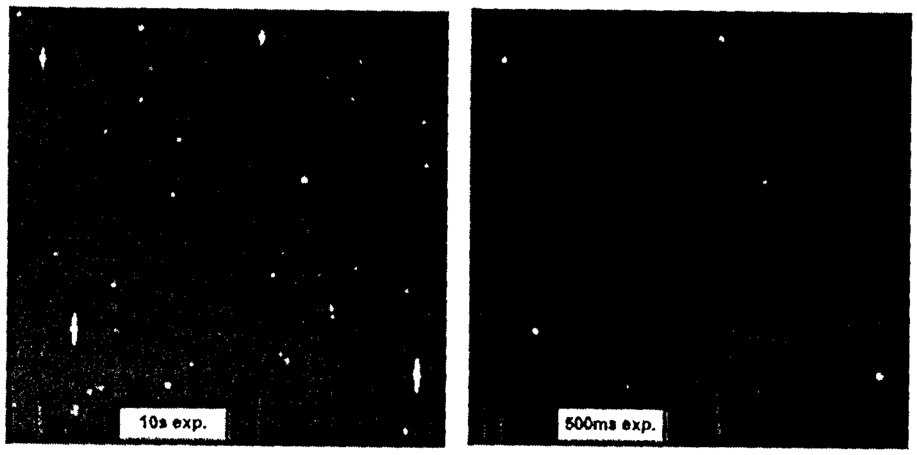
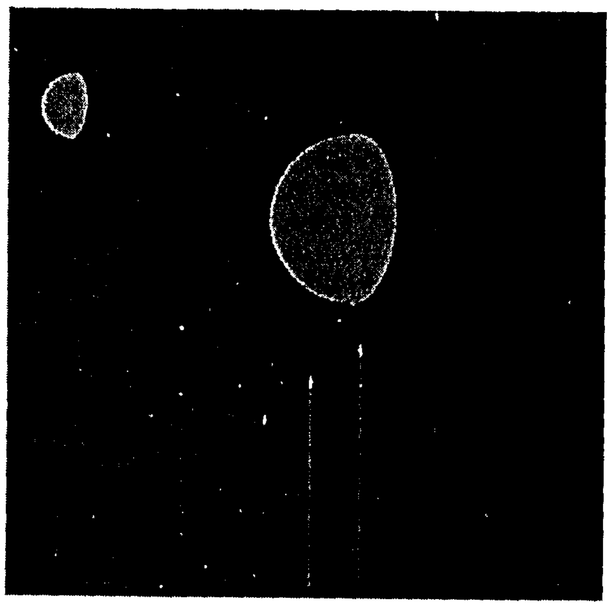


Figure 11: A comparison of the IEU pixel blooming model with lab test data. Although this picture can only show a gross resemblence, the fidelity of the model at the pixel level has been verified.



Extended Body Simulation

Since Cassini will be operating in the vicinity of Saturn and its moons, the possibility of tracked stars being occulted by an extended body exists. The IEU model captures only the gross effects important to star-tracking:

- A "body" consists of a sphere with 2 constant brightness regions separated by a terminator.
- The body may elm possess a ring, modeled as a flat washer with constant brightness.
- Up to 16 bodies are kept track of simultaneously.

The simulation does not:

- use the same full optical model as stars do
- motion-smear the extended body
- fully simulate readout smearing 01 the body; a fast but crude model is used
- allow bodies to bloom

The IEU snapshot at left shows an arbitrary arrangement of two bodies in front of the Orion constellation. Nde that the readout streak due to the star at the top of the frame properly overleps the larger body.

Figure 12: IEU extended body simulation example.

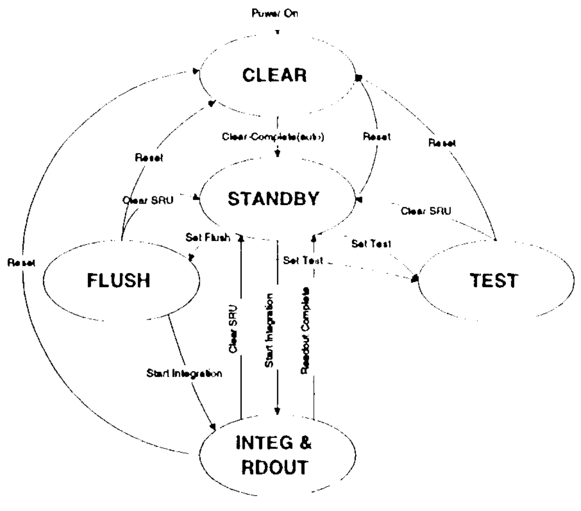
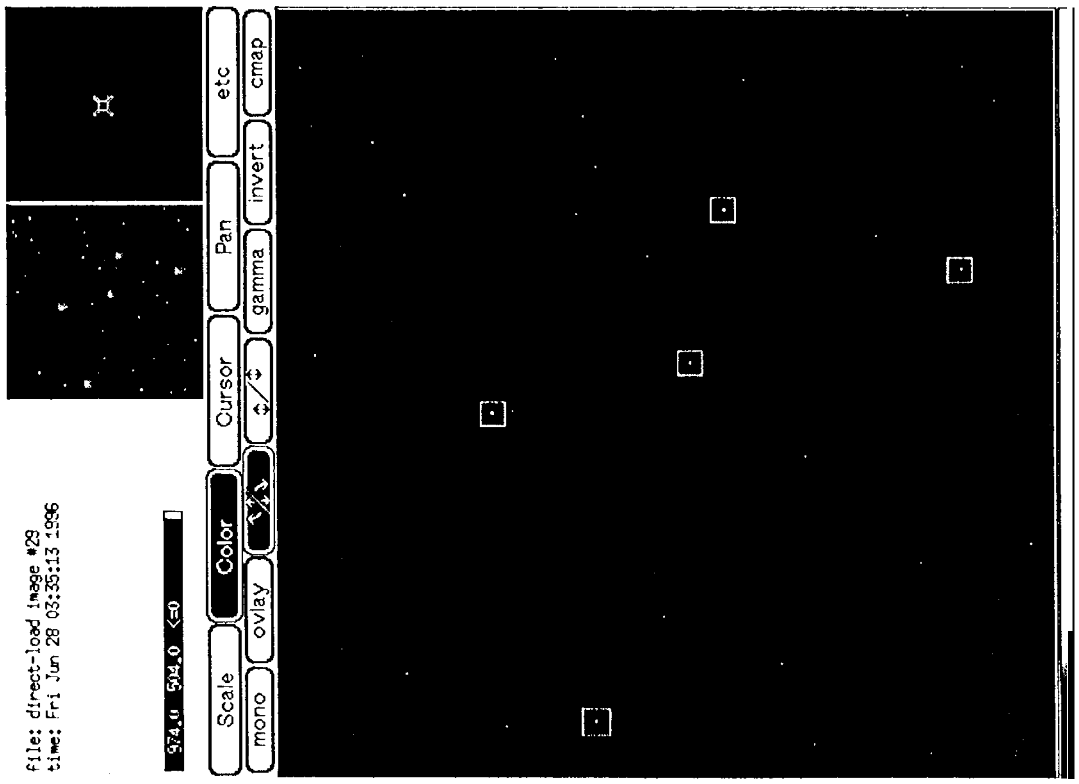


Figure 13: The SRU's state transition diagram. Normal operation involves only STANDBY and INTEGRATION & READOUT modes. The state machine logic is implemented in the IEU, although the functionality of the FLUSH and TEST modes are not.



```
file: direct-load image #29
time: Fri Jun 28 03:35:13 1996
```

Figure 14: Sample screen-shot of the IEU's SAOImage display, showing StarID tracking near Polaris. All zoom and pan features are available during a simulation, allowing detailed observation. Again the apparent "highlighting" of the readout windows is a result of noise being computed only for window contents, while the window borders are added specifically for the display.

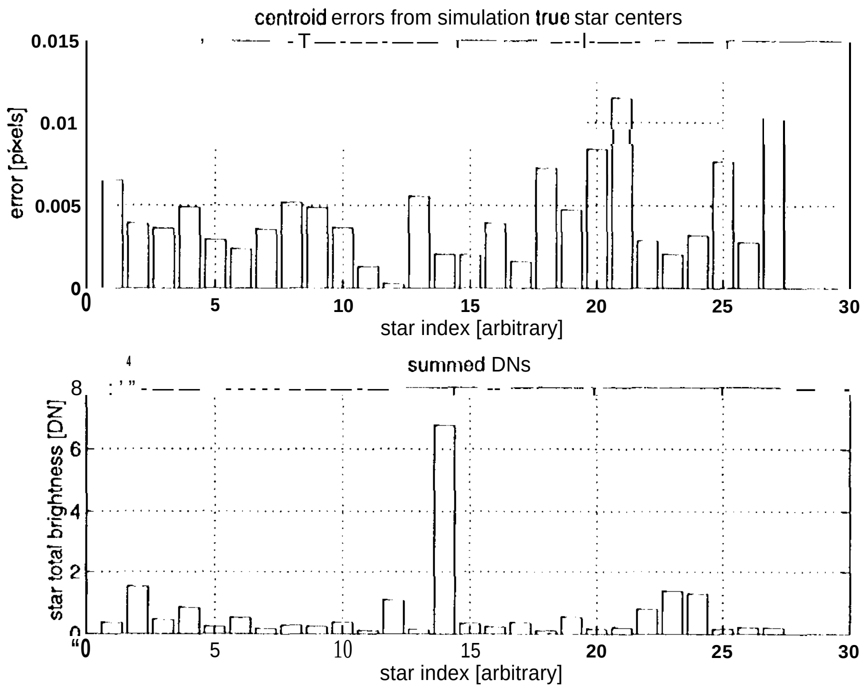


Figure 15: Analysis to verify that no significant position hiss is introduced in the IEU's PSF and motion-streak models. The errors are with respect to ideal star centers which are known to the simulation.

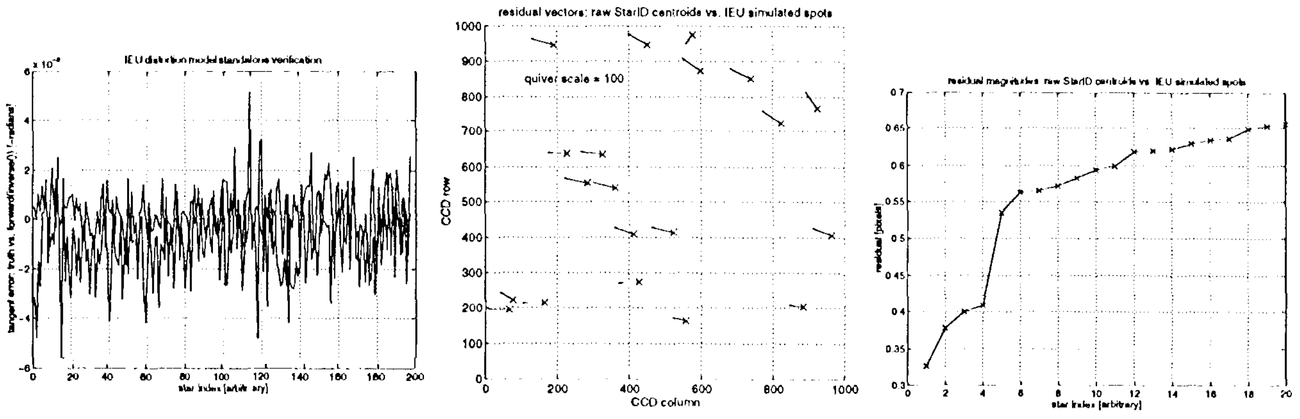


Figure 16: The leftmost figure shows verification of the IEU distortion inverse-model's accuracy. The middle and right figures show the magnitude and vector differences between the IEU's predicted spot locations and StarID's raw centroids. The estimated attitude is RA= 300.0765° , Dec= 10.01480 , Twist= 90.634°